**Objective**

We’re going to create an abstract base class with a couple of derived classes. Concepts we will be using / reinforcing in this lab:

* Rule of 3
* virtual methods
* inheritance
* vector class
* dynamic memory

**Base.h/cpp**

Create a Base class. This will be our ABC that will serve as our blueprint for our derived classes. Give the class a public DisplayRecord function and make it pure virtual (meaning it has no definition):

virtual void DisplayRecord() = 0;

Also add a private char\* member to the class for the name and initialize it to nullptr. Next, implement a public SetName method (it must do a deep copy of the memory since the name is a dynamic pointer). You may use the CopyString method from lecture.

Finally, fully implement the Rule of 3 for the class because it has a dynamic memory member.  
HINT: You only need to write the operator= and just have your copy constructor call that method (\*this = copy;) since they will usually have the same code.

**Employee.h/cpp**

Create an Employee class that publicly derives from Base. Give it a private int member for the salary and implement a public SetSalary method. Then override and implement the DisplayRecord method such that it first calls the base’s DisplayRecord method, then prints the salary to the screen.

**Student.h/cpp**

Same as the Employee class but instead of a salary give it a float member for the GPA.

**Main.cpp**

Declare a vector of Base pointers in global space above main. Then create a menu loop that will allow the user to add a record, display all records, duplicate a record, or exit the menu loop. Remember to delete all records in the vector at the end of the program. Next define the following functions so you can call them in your menu.

**AddRecord()**

Ask the user which type of record they want to add, an employee or a student. Once they have made their selection, dynamically create a new pointer of the selected type of record and have them fill out the record’s members via its Set methods, then add it to the vector using the vector method .push\_back().

**DisplayRecords()**

In this function, loop through the vector of records and call each pointer’s DisplayRecord() method.

**DuplicateRecord()**

Ask the user for an integer index for which record in the vector to create a copy of. Then use the dynamic\_cast technique from lecture to determine whether that record is a Student or Employee. Depending on its type, dynamically new another record of that same type and pass the user’s chosen record to the constructor call so that the copy constructor will be executed. Lastly, add the new record to the vector.